

# Adaptive Reconfiguration Moves for MCMC inference in Dirichlet Process Mixtures

**Tue Herlau**  
**Morten Mørup**  
**Mikkel N. Schmidt**  
*DTU Compute*  
*Technical University of Denmark*  
*Richard Petersens plads 31,*  
*2800 Lyngby, Denmark*

TUHE@DTU.DK  
 MMOR@DTU.DK  
 MNSC@DTU.DK

**Yee Whye Teh**  
*Department of Statistics*  
*University of Oxford*  
*1 South Parks Road,*  
*Oxford OX1 3TG, U.K.*

Y.W.TEH@STATS.OX.AC.UK

**Editor:** Unknown Editor

## Abstract

Bayesian mixture models are widely applied for unsupervised learning and exploratory data analysis. Markov chain Monte Carlo based on Gibbs sampling and split-merge moves are widely used for inference in these models. However, both methods are restricted to limited types of transitions and suffer from torpid mixing and low accept rates even for problems of modest size. We propose a method that considers a broader range of transitions that are close to equilibrium by exploiting multiple chains in parallel and using the past states adaptively to inform the proposal distribution. The method significantly improves on Gibbs and split-merge sampling as quantified using convergence diagnostics and acceptance rates. Adaptive MCMC methods which use past states to inform the proposal distribution has given rise to many ingenious sampling schemes for continuous problems and the present work can be seen as an important first step in bringing these benefits to partition-based problems.

## 1. Introduction

Mixture models are used for unsupervised learning and exploratory analysis to understand the structure in data by partitioning a set of observations into non-overlapping blocks. In this work we consider a Bayesian approach to probabilistic inference of partitions where a Dirichlet process is used as the prior for the partitions, and the goal is to estimate the posterior density of partitions using Markov Chain Monte Carlo sampling. Dirichlet process mixture models (Escobar and West, 1995; Antoniak, 1974; Lau and Green, 2007) have been applied to a wide range of problems including topic modeling (Teh et al., 2004), multi-task learning for classification (Xue et al., 2007), and relational data analysis (Kemp et al., 2006; Xu et al., 2006).

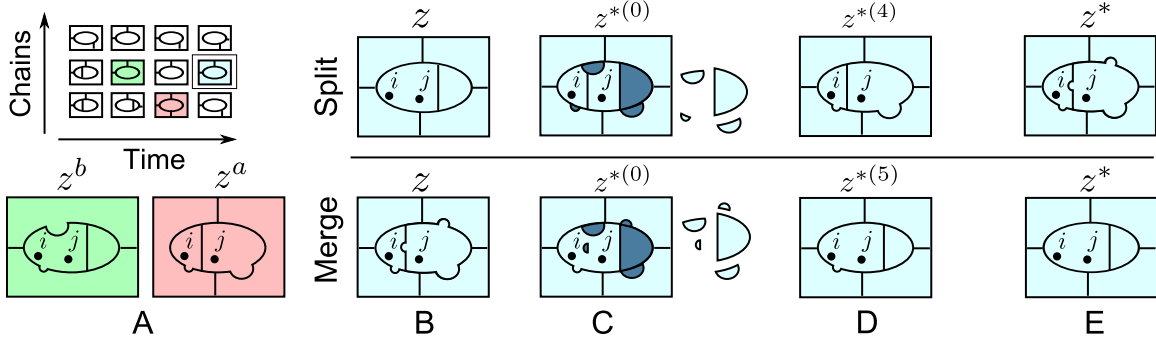


Figure 1: A single reconfiguration move applied to the partition  $z$ . Assume two partitions  $z^a$  and  $z^b$  as well as two vertices  $i, j$  has been chosen from the past states of  $S = 3$  chains. In (A) is shown the initial state  $z$ . by computing the coarsest common refinement between  $z, z^a$  and  $z^b$  the method construct the initial split and a set of blocks where the 3 partitions agree. These blocks are assumed initially removed from the partition (C). The removed blocks are Gibbs sampled into the problem (D) and finally all singleton elements (subject to certain restrictions) are allowed to move either from outside the blocks containing  $i, j$  and into the blocks containing  $i, j$ , or from inside the blocks containing  $i, j$  and out creating the final partition  $z^*$  in (E). The bottom row show the same process applied to create the reverse (merge) operation. Notice the total number of blocks in  $z, z^*$  remain 6.

Two common approaches to inference by MCMC are Gibbs and split-merge (SM) sampling (Neal, 1992; Jain and Neal, 2004). For conjugate models, Gibbs sampling iteratively assigns each observation to a set of candidate blocks corresponding to existing populated blocks or a new block. This makes Gibbs sampling easy to implement. However, as pointed out by Celeux et al. (2000) Gibbs sampling is prone to get stuck in local modes and significantly over- or underestimates the true number of blocks. This behavior stem from the incremental nature of Gibbs sampling which prevents the joint movement of observations.

This motivates the use of split-merge operations. Here, a split move consist of selecting a single block containing at least two observations and splitting the block into two new blocks thereby increasing the number of blocks by one (Green and Richardson, 2001; Dahl, 2003; Jain and Neal, 2004). A merge operation is the inverse procedure where two blocks are merged into one. In the simplest form the proposed split is made at random, however the chance of randomly selecting a favorable split can be very small and these moves will have high reject rate. Jain and Neal (2004) proposed using a more complex proposal distribution for a split-move obtained by applying a number of intermediate Gibbs updates to reach equilibrium states. In the following we will use split-merge to refer to the method of Jain and Neal (2004) unless otherwise stated.

While theoretically attractive, split-merge operations have some inherent drawbacks. These include (1) The problem of coupling: split-merge operations restrict themselves to only considering the variables contained in one or two blocks. Often it is the case no single split (or merge) of a block is favorable unless variables from other blocks are allowed to change configuration as well. (2) The problem of poor candidates: Very few blocks of variables should be either split or merged. For this reason, split-merge moves will most

of the time attempt to split or merge blocks where the configurations in which they are split (or merged) are highly improbable. (3) The problem of long walks to equilibrium: Split-merge attempt to reach equilibrium states by performing a large update and then slowly sampling towards equilibrium by changing a single variable at a time. While this will eventually reach equilibrium states, the reverse transition probability for a merge move may be very low and it requires many intermediate iterations.

We propose a new method for sampling partition-based models which attempt to address all the above issues (see figure 1 for an illustration). The primary goal is to overcome the limitation (1) by allowing observations to travel between the blocks currently being split or merged and the other blocks of the partition. For instance figure 1 illustrates the situation where a subset of a block of variables is being moved from one block to another thereby conserving the total number of blocks. As a result, our method does not require the number of blocks to increase or decrease deterministically, but can in principle change in any direction, this allows for a far larger set of proposal states.

Where split-merge created proposal states near equilibrium through restricted Gibbs sweeps, our method make use of adaptive Markov-Chain Monte Carlo (Atchadé and Rosenthal, 2005; Roberts and Rosenthal, 2007) to overcome limitation (2) and (3). We evaluate multiple chains in parallel and use disagreement between assignment of vertices between the chains to ensure blocks are only either split or merged if there is disagreement between the chains on their status. In addition, the multiple chains are used to determine the blocks of variables partitions from different chains agree upon and move these blocks jointly, avoiding the many intermediate restricted Gibbs operations in (3).

Due to the adaptive nature of the method and the availability of moves beyond split-merge we dub the method Adaptive Reconfiguration Moves (ARM).

In terms of demands on the model, ARM require the ability to compute change in likelihood when a set of variables are all reassigned at once in a Gibbs sweep. However, for all models we are aware of this change in implementation is a relatively straight-forward generalization of the methods required for Gibbs sampling.

For simplicity we have chosen to focus only on the discrete sampling problems similar to those considered by Jain and Neal (2004), and we will assume the models allow the infinite-dimensional parameters to be integrated out analytically. We evaluate the sampling procedure on two examples of partition-based problems, the Bernoulli mixture model (cf. (Toutenburg, 1985)) also considered by Jain and Neal (2004), and the Infinite Relational Model of Kemp et al. (2006) for undirected graphs.

Sampling operations where blocks are being reassigned also form the basis of the Swendsen-Wang method from statistical physics Swendsen and Wang (1987). Swendsen-Wang consider the graph-partitioning problem for two classes and create proposal moves where groups of vertices may change partition by considering disconnected components obtained by stochastically thinning the edges. The generalized Swendsen-Wang algorithm of Barbu and Zhu (2005) generalize the Swendsen-Wang method to partitions with more than two blocks, however it is still formulated as edge-thinning problem and not obviously amendable to the situation considered herein.

### 1.1 Partition-based models

The Dirichlet process is a prior over measures and is a popular way to induce a prior over random partitions (Ferguson, 1973; Antoniak, 1974). The Dirichlet process admits several equivalent formulations the most straight-forward of which is the stick-breaking representation of Sethuraman (1991). For a random measure  $H$  and concentration parameter  $\alpha$  it is given as the generative process

$$v_k \sim \text{Beta}(1, \alpha) \quad \theta_k \sim H \quad (1a)$$

$$\beta_k = v_k \prod_{\ell=1}^{k-1} (1 - v_\ell) \quad G = \sum_{k=1}^{\infty} \beta_k \delta_{\theta_k} \quad (1b)$$

which we write  $G \sim \text{DP}(\alpha, H)$  for the random measure  $G$  and  $\beta \equiv (\beta_k)_{k=1}^{\infty} \sim \text{GEM}(\alpha)$  for the induced distribution over the weights  $\beta_k$  (Pitman et al., 2002). Consider a set of  $n$  observations  $\{y_1, \dots, y_n\}$  and assume the following generative model

$$G \sim \text{DP}(\alpha, H) \quad (2a)$$

$$\text{for } i = 1, \dots, n, \quad \theta_i | G \sim G \quad (2b)$$

$$y_i | \theta_i \sim F(\theta_i) \quad (2c)$$

where  $F$  is the distribution of  $y$  conditional on the parameter  $\theta$ . The  $K$  unique values of  $(\theta_i)_{i=1}^n$ , denoted  $(\theta_k^*)_{k=1}^K$ , induce a partition over  $[n] = \{1, \dots, n\}$  through the equivalence relation. In particular for each  $k = 1, \dots, K$  define  $B_k \equiv \{i : \theta_i = \theta_k^*\}$ , then the collection of sets  $B_k$ , written as  $z = (B_1, \dots, B_K)$ , form a partition of  $[n]$  and each  $B_k$  is called a *block* of the partition. Since the  $\theta_i$  in eq. (1) are random, this induces a random partition of  $[n]$ . The distribution of this random partition is called a *Chinese restaurant process* characterized by  $\alpha$  and has the density

$$p_{\text{CRP}}(z | \alpha) = \frac{\Gamma(\alpha) \alpha^K}{\Gamma(\alpha + n)} \prod_{k=1}^K \Gamma(|B_k|) \quad (3)$$

where  $|B_k|$  denote the number of elements in a block  $B_k$ , see Pitman et al. (2002) for addition details.

We will in the following consider models based on partitions induced by a Dirichlet process and which admit a particular analytical simplification discussed below. In the simulations we will consider vectorial data of the form  $y_i$ ,  $i = 1, \dots, n$  and relational data consisting of symmetric matrices  $y_{ij}$ ,  $1 \leq i < j \leq n$ , both with representations which may be written as:

$$\begin{array}{llll} \beta \sim \text{GEM}(\alpha) & z^i | \beta \sim \text{Mult}(\beta) & 1 \leq i \leq n \\ \text{Mixture models: } \theta_k^* \sim H, & 1 \leq k \leq K & y_i \sim F(\theta_{z^i}^*) & 1 \leq i \leq n \\ \text{Relational models: } \theta_{k\ell}^* \sim H, & 1 \leq k < \ell \leq K & y_{ij} \sim F(\theta_{z^i z^j}^*) & 1 \leq i < j \leq n \end{array}$$

where  $z^i$  is the index  $k$  such that  $i \in B_k$  for a partition  $z = (B_1, \dots, B_K)$  and with the convention  $\theta_{\ell k}^* = \theta_{k\ell}^*$ . Notice this representation for the mixture model is equivalent to the

generative procedure in eq. (2). The simplification we will assume is the distributions  $F$  and  $H$  are conjugated such that all  $\theta^*$ -parameters can be marginalized out analytically and we will call such a model *conjugate* in the following. Performing this marginalization leave us with a posterior of the form

$$p(Y, z) = p(Y|z)p_{\text{CRP}}(z) \equiv q(z) \quad (4)$$

where  $Y$  is the data (a vector or matrix). As indicated, we will in the following abbreviate the function  $p(Y, z)$  by  $q(z)$  and call the above a *model* for partitions.

## 1.2 Gibbs sampling

The methods discussed in this paper are easiest described in notation which admit an ordering of the sets in the partition. Accordingly, the basic object will be a list of sets written  $z = (B_1, B_2, \dots, B_K)$  where each  $B_\ell$  is denoted a *block* of the partition, however we will in general use the word block to refer to a general subset.

Let  $|z|$  define the number of non-empty blocks of  $z$  and  $\cup z = \cup_{k=1}^{|z|} B_k$  all observations contained in  $z$ . If each  $B_k \neq \emptyset$  and for all  $k \neq \ell$ :  $B_\ell \cap B_k = \emptyset$  we will say  $z$  is a *partition* of  $X = \cup z$ .

For a list  $z = (B_1, \dots, B_K)$  we let  $z(k)$  denote the  $k$ 'th block of  $z$ , i.e.  $z(k) = B_k$ . If in addition  $z$  is a partition of  $X$  and  $i \in X$  is any observation, we denote by  $z_i$  the unique *block* containing  $i$ :

$$z_i \equiv B_j \text{ such that } i \in B_j$$

In addition, if  $B \in z$  is a block in  $z$ , denote by  $|B|$  the number of elements in  $B$ . For all  $h \leq |B|$  let  $B(h)$  indicate the  $h$ 'th value of  $B$  in ascending order, specifically  $B(1) = \min B$ .

We also define common operations on the partitions. For a block  $A$ , we let  $z \setminus A$  denote the partition  $z'$  obtained by removing the set  $A$  from each block of  $z$  as well as any empty sets. Specifically:

$$z' = \{B_k \setminus A : B_k \in z \text{ and } B_k \setminus A \neq \emptyset\}$$

For two lists of blocks  $z^a, z^b$  we let  $z = z^a \cup z^b$  denote the effect of concatenating  $z^b$  after  $z^a$ . Formally let  $m = \max\{k : |z^a(k)| > 0\}$  be the last non-empty set of  $z^a$ , then  $z$  is the list such that  $z(k) = z^a(k)$  for  $k \leq m$  and  $z(k) = z^b(k - m)$  for  $k > m$ .

Next, for a model of partitions  $z$  on a space  $X$  (by which we mean a function  $q(\cdot)$  as described in eq. (4)) we define the notion of a (restricted) Gibbs sweep of a non-empty block  $C \subset X$ . A restricted Gibbs sweep is the operation where all variables in the block  $C$  are jointly reassigned to a block from a fixed set of available blocks, the particular block being chosen with probability proportional to the likelihood as defined by  $q(z)$ .

Specifically, for a partition  $z = (B_1, B_2, \dots, B_K)$  assume  $I$  is a list of blocks  $I = (A_1, A_2, \dots, A_m)$  such that for any  $A_i$  either  $A_i = \emptyset$  or else there exist a  $j$  such that  $A_i = B_j$ . We will also assume  $C$ , the observations which will be moved, are either contained in a single block of  $z$  or disjoint from the observations partitioned by  $z$ . Formally there either exist  $B \in z$  such that  $C \subset B$  or  $C \cap (\cup z) = \emptyset$ . If these conditions are met we define by

$$(z^*, \pi_k^*) = \text{sweep}_{C,I}(z) \quad (5)$$

the operation wherein the observations in  $C$  are assigned into one of the blocks in  $I$  according to the likelihood. Specifically for each  $A_k \in I$  we define a new partition  $z^{(k)}$  and weight  $\pi_k$  as:

$$z^{(k)} = (z \setminus (C \cup A_k)) \cup (C \cup A_k)$$

$$\pi_k = \frac{q(z^{(k)})}{\sum_{m=1}^{|I|} q(z^{(m)})}.$$

Notice in non-list notation the partition  $z^{(k)}$  is simply

$$\{B_\ell \setminus (A_k \cup C) : B_\ell \in z \text{ and } B_\ell \setminus (A_k \cup C) \neq \emptyset\} \cup \{A_k \cup C\}$$

and  $z^*$  is obtained as the partition  $z^{(k)}$  where  $k$  is choosen randomly from the catagorial distribution with weights  $(\pi_k)_{k=1}^{|I|}$ . We will also write

$$(z^*, \pi_k) = \text{sweep}_{C,I}(z | z_{C(1)} = k) \quad (6)$$

as the operation of (forcibly) assigning the variables  $C$  into block  $k$ , i.e., choosing  $z^* = z_{(k)}$  and letting  $\pi_k$  be the corresponding probability. The standard Gibbs sweeps considered by MacEachern (1994) and Neal (2000) is obtained when  $|C| = 1$  and  $I$  contain the non-empty blocks of  $z$  and an empty set. In order to more easily acommodate this case we use the simplified notation

$$(z^*, \pi_k) = \text{sweep}_C(z) \quad (7)$$

for the situation  $I = z \cup \{\emptyset\}$  in eqn. (5).

### 1.3 split-merge sampling

The incremental nature of Gibbs sampling makes it prone to get stuck in local modes where it either over or under estimates the true number of blocks Celeux et al. (2000). Metropolis-Hastings proposal moves are a popular supplement to Gibbs sampling in that it provides a flexible framework to construct bolder update moves based on domain knowledge Metropolis et al. (1953); Hastings (1970). The Metropolis-Hastings algorithm samples from a distribution  $q$  by first drawing a candidate state  $z^*$  according to a proposal density  $T_\phi(z^*|z)$  parameterized by  $\phi \in \Phi$  and setting the next state of the chain equal to the candidate state with probability

$$a(z^*, z) = \min \left[ 1, \frac{T_\phi(z|z^*)}{T_\phi(z^*|z)} \frac{q(z^*)}{q(z)} \right] \quad (8)$$

Otherwise the new state remains the current state  $z$ . The parameters  $\phi$  can either be generated deterministically or stochastically and will be discussed later, in the particular case of split-merge it will consist of two observations  $i, j$ .

A series of states generated by proposing and accepting according to eq. (8) will leave the distribution invariant and will sample the problem  $q$  provided the chain is ergodic. One particular set of proposal moves is split-merge moves where either a single block is split into

---

**Algorithm 1** split-merge sampling by Jain and Neal (2004)
 

---

- 1: **Construct the launch state**  $z^{(l)}$ : Remove all elements in the block(s) containing  $i$  and  $j$ . If  $z_i = z_j$  perform a split otherwise a merge operation. In either case initialize  $i$  and  $j$  in separate clusters.

$$z^{(l)} \leftarrow z \setminus \{z_i \cup z_j\}, \quad (9)$$

$$z^{(l)} \leftarrow z^{(l)} \cup \{\{i\}, \{j\}\}. \quad (10)$$

Randomly assign the missing observations  $S = z_i \cup z_j \setminus \{i, j\}$  between the two new blocks  $z_i^{(l)}, z_j^{(l)}$ .

- 2: **Perform  $L$  restricted Gibbs sweeps on**  $z^{(l)}$ : Each Gibbs operation iterates over the variables  $h \in S$  and sample  $h$  between the blocks containing  $i$  and  $j$

$$I \leftarrow (z_i^{(l)}, z_j^{(l)}), \quad (11)$$

$$z^{(l)} \leftarrow \text{sweep}_{h,I}(z^{(l)}). \quad (12)$$

- 3: **if**  $z_i \neq z_j$  **then**

- 4:   **Merge**: Let  $z^*$  be the partition with the two blocks containing  $i$  and  $j$  merged

$$z^* \leftarrow (z \setminus \{z_i \cup z_j\}) \cup \{z_i \cup z_j\}.$$

Compute the reverse transition probability *from the launch state* by performing one single restricted Gibbs sweep over the observations  $h \in S$  forcing the observations to take the same assignment as in  $z$

$$I \leftarrow (z_i^{(l)}, z_j^{(l)}) \quad (13)$$

$$(\cdot, \pi^h) \leftarrow \text{sweep}_{h,I}(z^{(l)} | z_h^{(l)} = z_i^{(l)} \text{ if } z_h = z_i \text{ otherwise } z_h^{(l)} = z_j^{(l)}). \quad (14)$$

Calculate the proposal probability  $T(z|z^*) = \prod_{h \in S} \pi^h$ .

- 5: **else if**  $z_i = z_j$  **then**

- 6:   **Split**: Otherwise let  $z^* = z^{(l)}$  and perform one final restricted sweep over all  $h \in S$

$$I \leftarrow (z_i^*, z_j^*), \quad (15)$$

$$(z^*, \pi^h) \leftarrow \text{sweep}_{h,I}(z^*). \quad (16)$$

Calculate the proposal probability  $T(z^*|z) = \prod_{h \in S} \pi^h$ .

- 7: **end if**
-

two new blocks or two blocks is merged into a single block. While the merge step is unique, there are multiple ways to perform the split step. One of the most popular is the split-merge method of Jain and Neal (2004). The method propose a split configuration by randomly selecting two observations  $i, j$  (ie. the background information consist of  $\phi = (i, j)$ ) then randomly distributing the observations assigned to the block(s) containing  $i, j$  between two new blocks, then perform a number of Gibbs updates restricted to only moving observations between these two new blocks to obtain near equilibrium split configuration and a final Gibbs update to get a Split-Proposal. It is only the last restricted Gibbs sweep which is used to compute the transition probability  $T(z^*|z)$  for a split (in this case the verse probability  $T(z|z^*) = 1$ ) or  $T(z|z^*)$  for a merge (in which case  $T(z^*|z) = 1$ ). Conditional on randomly selected observations  $i \neq j$  the construction is listed in algorithm 1.

Convergence of the method can be seen either by considering an augmented target space, or by considering the launch state  $z^{(l)}$  as well as the vertices  $i, j$  as indices in a very large set of transition kernels which are selected stochastically according to the above procedure (Jain and Neal, 2004; Tierney, 1994). A slight variation of the above method discussed by Dahl (2003) is obtained by simply omitting the step where the elements are randomly assigned to the blocks of  $z^{(l)}$  containing  $i$  and  $j$ , however we have not found significant improvement with this variation for the considered problems.

## 2. Proposed Method

While thermalization of the initial (random) split through restricted Gibbs sweeps can be expected to improve the launch state the method is still limited in two ways: Firstly, based on experimentation with the Infinite Relational Model we found that even when sampling to equilibrium it was often the case there was favorable split or merged configurations of two observations  $i, j$ , however no single split (or merge) operation of two blocks could reach the more favorable state without altering the assignment of vertices assigned to other blocks. Secondly, merge moves will tend to have low accept rates unless only a single split configuration is favorable. To illustrate this, suppose the current split configuration is different from that found by performing restricted Gibbs moves on the launch state. Then the final (forced) Gibbs sweep will have to (forcibly) perform a large number of re-assignments to transform the one split-configuration into the other which may often be energetically unfavorable.

As outlined in figure 1, our proposed method allows observations to not only travel between the two blocks containing  $i$  and  $j$ , but also from blocks not containing  $i$  and  $j$  and into blocks containing  $i$  and  $j$  and vice versa. To retain tractability when computing the transition probabilities  $T$  we restrict the method to *not* allow observations to travel *between* blocks *not* initially containing  $i$  and  $j$ . The corresponding space of possible transitions is larger than for a Split-Merge operation, and a single restricted Gibbs sweep cannot be expected to produce equilibrium states. We overcome this difficulty by re-using past information of which observations tend to be grouped together and can therefore be expected to change block assignments together. This information is obtained by evaluating multiple chains in parallel and use their agreement or disagreement to both select  $i, j$  as well as construct the relevant blocks, the idea being that if a set of observations are contained in the same block in both a split and merged configuration it makes sense to update them



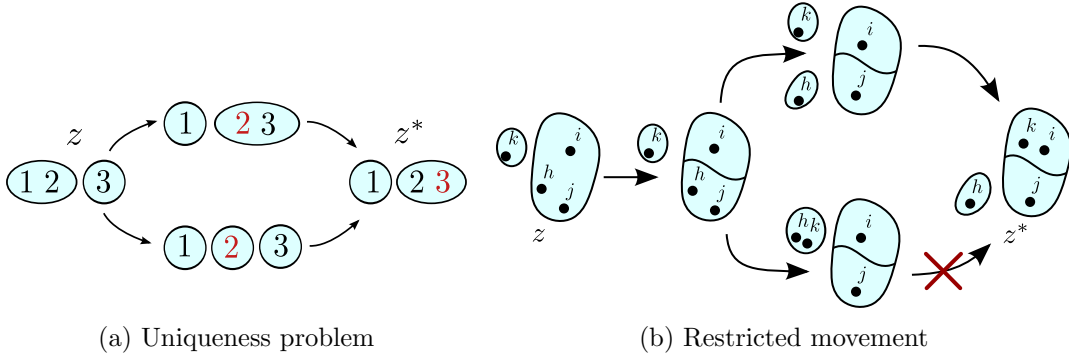


Figure 2: *Left:* Problem defining a unique sampling path for a simple 3-variable problem. The final partition  $z^* = \{\{1\}, \{2, 3\}\}$  is obtained from the initial partition  $z = \{\{1, 2\}, \{3\}\}$  by Gibbs sampling observation 2, 3. This however allows for two paths and thus the transition probability  $T(z|z^*)$  is not the product of each Gibbs transition probability. *Right:* A realistic uniqueness problem which arise in our method. The observations  $h$  and  $k$  may arrive at their final position in two ways. To ensure uniqueness, if a block has had variables sampled into it (such as  $k$ ), we disallow all *original* variables to leave the block, thereby removing the bottom path.

jointly. Convergence of this scheme is guaranteed under the Adaptive MCMC framework, see Roberts and Rosenthal (2009, eqs. (1.1)).

We will first introduce a simplified version of the method denoted Simplified Reconfiguration Moves (sRM) which does not involve movement of blocks or adaptive MCMC.

## 2.1 Simplified Reconfiguration Moves

Denote by  $z$  the current state of the chain and let  $z^*$  denote the next state of the chain. Similar to the split-merge algorithm we assume the transition kernels are selected from a set of random kernels  $T_\phi(z^*|z)$ ,  $\phi$  again being a set of index parameters. The method construct  $z^*$  through a number of Markov steps  $z^*(m)$  indexed by  $m$ , such that  $z^*(0)$  is constructed deterministically from the initial state and  $z^*(M)$  ( $M$  being defined later) corresponds to  $z^*$ .

While it is easy to compute the Markov transition probabilities  $T_z^{(m)}$  for each step of nearly any construction, the difficulty is to ensure their product corresponds to  $T(z^*|z)$ . The main problem being to ensure the path from  $z$  to  $z^*$  is unique. Consider for instance the case of a partition of a set of  $n = 3$  elements,  $z = \{\{1, 2\}, \{3\}\}$ , and assume a proposal kernel is constructed by first Gibbs sampling element 2 and then 3. This does not define a unique path to the final configuration  $z^* = \{\{1\}, \{2, 3\}\}$  (see figure 2a for an illustration of the two paths) and the transition probability is not simply the product of the transition probability of each Gibbs kernel. For the Split-Merge method uniqueness was ensured since in the final configuration in e.g. a split move, each observation is either in the same block as  $i$  or  $j$  and the choice is unique, however for the proposed method in which observations can travel between many blocks and blocks can be created or destroyed more care is required.

---

**Algorithm 2** Simplified Reconfiguration Moves
 

---

```

1: Initialize  $T_z \leftarrow 1$ .
2: If  $z_i = z_j$  perform a split otherwise merge move.
3: Remove from  $z$  the blocks  $z_i$  and  $z_j$  containing  $i, j$ :  $z^* \leftarrow z \setminus \{z_i \cup z_j\}$ .
4: Add  $i, j$  to  $z^*$ . If splitting:  $z^* \leftarrow z^* \cup \{\{i\}, \{j\}\}$  else if merging:  $z^* \leftarrow z^* \cup \{\{i, j\}\}$ .
5: for Each observation  $h$  in  $\{s : s \neq i, j\}$  do
6:   if  $h \in z_i \cup z_j$  then
7:     Perform an unrestricted Gibbs move of  $h$  and update the transition probabilities
       
$$(z^*, \pi_k) \leftarrow \text{sweep}_h(z^*) \quad \text{and} \quad T_z \leftarrow \pi_k T_z$$

8:   else if  $h \notin z_i \cup z_j$  then
9:     if  $z_h \cap z_h^* = \{h\}$  and  $|z_h^*| \geq 2$  then
10:      If other observations has been sampled into  $h$ 's block in  $z^*$ , and  $h$  is the last of the original
        observations from  $z$ , then do not allow  $h$  to change assignment (see eq. (17)):  $I \leftarrow (z_h^*)$ .
11:    else
12:      Allow  $h$  to either stay or move into the block(s) containing  $i, j$ .
        If splitting:  $I \leftarrow (z_i^*, z_j^*, z_h^*)$  otherwise:  $I \leftarrow (z_j^*, z_h^*)$ 
13:    end if
14:    Perform a Gibbs sweep restricted to the blocks  $I$ :
       
$$(z^*, \pi_k) \leftarrow \text{sweep}_{I,h}(z^*) \quad \text{and} \quad T_z \leftarrow \pi_k T_z.$$

15:  end if
16: end for
    
```

---

### 2.1.1 UNIQUENESS OF SAMPLING PATH

Consider the case outlined in Figure 2b corresponding to a split-move of observations  $i, j$ . In the top-path observation  $h$  form a new singleton block and then another singleton block  $k$  enter the block containing observation  $i$ . In the bottom path observation  $h$  join the singleton block formed by observation  $k$ , then  $k$  attempt to join the block containing  $i$ . Since the final partition is the same this creates a non-unique path from  $z$  to  $z^*$ . To disallow this possibility we will impose the restriction if, in the course of a proposal move, a new observation (such as  $h$ ) enters a block, *all* observations originally assigned to this block cannot leave it (such as  $k$ ). Symbolically this restriction corresponds to the case where

$$\begin{aligned}
 & z_h^* \cap z_h = \{h\} && (h \text{ is the last of the original observations remaining}) \\
 \text{and} \quad & |z_h^*| \geq 2 && (z_h^* \text{ contain other observations than the original}) . \quad (17)
 \end{aligned}$$

This along with the observation that for two blocks  $A, B$  *not* containing  $i, j$  variables cannot move from  $A$  to  $B$  or vice versa (see eq. (17)) is sufficient to ensure uniqueness. However, we will return to this point after giving the full method. The simplified proposal distribution can be seen as algorithm 2.

## 2.2 Full method

The full method is obtained by including joint updates of blocks of observations in algorithm 2 corresponding to plate (C) in figure 1. These blocks are obtained by including, in addition to  $i, j$ , two partitions  $z^a$  and  $z^b$  as background information  $\phi$  under the restriction  $z_i^a = z_j^a$  and  $z_i^b \neq z_j^b$ . The list of candidate blocks to update jointly are obtained from the

coarsest common refinement of  $z^a, z^b$  and  $z$  *restricted* to the set of observations in the same block as  $i, j$ ,  $z_i \cup z_j$ . Recall the coarsest common refinement of two partitions  $z, z'$  is defined as the partition

$$z \vee z' = \{A \cap B : A \in z, B \in z', A \cap B \neq \emptyset\}.$$

The intuitive notion is that if variables are assigned similarly in all three blocks then the split (or merge) operation of  $i$  and  $j$  to obtain the new state  $z^*$  will likely leave these consistent assignments invariant as well, see figure 1 for an illustration of a single move.

---

**Algorithm 3** Adaptive Reconfiguration Move
 

---

```

1: Initialize  $T_z \leftarrow 1$  and compute the coarsest common refinement  $c \leftarrow z^a \vee z^b \vee z$ .
2: If  $z_i = z_j$  perform a split otherwise a merge move.
3: Remove from  $z$  the blocks  $z_i$  and  $z_j$  containing  $i, j$ :  $z^* \leftarrow z \setminus \{z_i \cup z_j\}$ .
4: Add blocks from  $c$  containing  $i, j$  to  $z^*$ . If splitting:  $z^* \leftarrow z^* \cup \{c_i, c_j\}$  else if merging:  $z^* \leftarrow z^* \cup \{c_i \cup c_j\}$ .

5: Let  $g'$  denote all elements of  $c$  not currently placed:  $g' \leftarrow c \setminus (\cup z^*)$ .
6: for each  $a$  in  $g'$  do
7:   Perform an unrestricted Gibbs move of  $a$  and update the transition probabilities
      
$$(z^*, \pi_k) \leftarrow \text{sweep}_a(z^*) \quad \text{and} \quad T_z \leftarrow \pi_k T_z.$$


8: end for
9: for each observation  $h$  not currently updated, ie. in  $\{s : s \neq i, j \text{ and } s \neq a(1) \text{ for all } a \in g'\}$  do
10:   if  $h \in z_i \cup z_j$  then
11:     Perform an unrestricted Gibbs move of  $h$  and update the transition probabilities
        
$$(z^*, \pi_k) \leftarrow \text{sweep}_h(z^*) \quad \text{and} \quad T_z \leftarrow \pi_k T_z.$$


12:   else if  $h \notin z_i \cup z_j$  then
13:     if  $z_h \cap z_h^* = \{h\}$  and  $|z_h^*| \geq 2$  then
14:       Implement the uniqueness constraint of eq. (17):  $I \leftarrow (z_h^*)$ .
15:     else
16:       Allow  $h$  to either stay or move into the block(s) containing  $i, j$ .
       If splitting:  $I \leftarrow (z_i^*, z_j^*, z_h^*)$  otherwise:  $I \leftarrow (z_j^*, z_h^*)$ 
17:     end if
18:     Perform a Gibbs sweep restricted to the blocks  $I$ :
        
$$(z^*, \pi_k) \leftarrow \text{sweep}_{I,h}(z^*) \quad \text{and} \quad T_z \leftarrow \pi_k T_z.$$


19:   end if
20: end for
    
```

---

To not restrict the move class, and since the common coarsest refinement of  $z, z^a, z^b$  and  $z^*, z^a, z^b$  may be different, we allow variables which have been moved as part of a block to be moved independently later. To avoid multiple-path issue we need to ensure each variable can only be updated once. When moving blocks this is ensured by treating the first element of a given block  $A$ ,  $A(1)$ , as an "earmark" of the block and the other variables  $A(2), A(3), \dots$  may then later be updated independently of the rest in a similar fashion as algorithm 2. In other words, when a block  $A$  is moved the probabilities in it's Gibbs sweep is computed based on the full likelihood, and since the blocks are constructed to be subsets of  $z_i$  and  $z_j$  the range of transition probabilities is the full Gibbs move in eq. (5). However when the other observations of  $A$ , for instance  $A(2)$  is later updated, we compute the available blocks for  $A(2)$  *not* based on it's current position (which may be outside  $z_i^*$

and  $z_j^*$ ), but again as a full Gibbs sweep since it's original configuration was with  $z_i$  and/or  $z_j$ .

In similar vein to the discussion of Jain and Neal (2004); Tierney (1994) we are free to choose the background information  $\phi$  deterministically or stochastically. In our approach we will consider a more general setting where  $\phi$  depend on the past history of the current chain and other chains, specifically by selecting initial split/merged configurations  $z^a$  and  $z^b$  which are used to construct the proposal from the past history of the chain. Since these states are selected at random from a growing set of past states, the distribution over pairs will converge and according to the theory of adaptive MCMC we will sample the correct stationary distribution (Atchadé and Rosenthal, 2005; Roberts and Rosenthal, 2007). A full description of how the algorithm propose a new configuration  $z^*$  and compute the transition probability  $T_z$  in eq. (8) conditional on  $i, j, z^a, z^b$  is given in algorithm 3.

### 2.3 Comments on convergence

We show different paths in the construction of  $z^*$  result in different final values of  $z^*$ , ie. the construction is unique allowing us to identify  $T_z$  with the transition probability  $T(z^*|z)$ . Since the initialization is deterministic the proof proceed by considering each iteration of the for loops in line 6 and 9 of algorithm 3 and line 5 of algorithm 2 in turn.

For a particular iteration  $m$ , let  $A$  be the block of observations currently being Gibbs sampled. If  $A \not\subseteq z_i \cup z_j$  then in the final configuration  $z^*$ ,  $A(1)$  (in fact  $A$  is a singleton set in this case) will either be the same block as  $i, j$  or in a different block than both  $i$  and  $j$ . As a result, this branch is unique.

Alternatively, if  $A \subseteq z_i \cup z_j$ ,  $A(1)$  may be assigned to a full set of blocks  $z^*$  as well as a new block. As before, if  $A(1)$  is assigned to the block(s) containing  $i$  and  $j$  it will remain with  $i$  (or  $j$ ) in the final value of  $z^*$  making this choice of assignment unique. Accordingly, we only need to consider configurations where  $A(1)$  is not assigned to to the block(s) containing  $i$  and  $j$ :

For each such existing candidate block  $B_k \in I$ , by the non-emptying condition that not all elements of  $B_k$  can later be removed: either because they have been assigned to  $B_k$  during past iterations of the method (and therefore cannot change assignment later) or if they were in  $B_k$  due to their initial assignment in  $z$  they cannot all leave due to the non-emptying condition in eq. (17).

In either case there exist elements of each set  $B_k$  which are different from  $i, j$  and such that  $A(1)$  will remain with these elements in the final partition  $z^*$  or, if  $A$  is assigned to a new block, there is no way for any elements outside of  $z_i$  and  $z_j$  to end up with  $A(1)$ . In either case the branch is unique as well.

Finally, to allow a growing number of past states, notice the probability of choosing any two initial states  $z^a, z^b$  changes proportionally to the inverse of the total number of past states. Since this rate converges to zero, the diminishing adaption condition of Roberts and Rosenthal (2009, eqs. (1.1)) is satisfied guaranteeing convergence.

### 2.4 Remarks

To finalize the description of the method we need to specify how the initial information,  $z^a, z^b$  and  $i, j$  was chosen. Our method evaluated  $S$  chains in parallel, such that  $Z^{st}$  correspond

to the state of chain  $s$  at time  $t$ . When sampling the next state,  $Z^{s(t+1)}$ , we selected  $z^a$  and  $z^b$  from the set of  $S[t/2]$  chains

$$\{Z^{s't'} | s' = 1, \dots, S \text{ and } t' = \lfloor t/2 \rfloor, \dots, t\}$$

at random under the constraints:  $z^a \neq z^b$  and one of the chains was selected from the subset where  $s' = s$ , i.e., the past history of the current chain  $s$ . Other choices are possible such as using the likelihood of previous states as a weight.

Having selected  $z^a$  and  $z^b$  we randomly select  $i, j$  from the set of all pairs where the two partitions disagreed:  $\delta_{z_i^a, z_j^a} \neq \delta_{z_i^b, z_j^b}$  and relabel  $z^a$  and  $z^b$  if  $z_i^a \neq z_j^a$  to agree with our conventions.

The description of ARM and sRM is not fully defined without specifying the order in which the lists of candidate blocks (or variables) are iterated over in line 5 of Algorithm 2 and line 6 and 9 of Algorithm 3. In the simulations we choose to iterate over the lists according to the size of the blocks (in descending order) and in case of equal size, according to the value of the first element in each block,  $A(1)$  (in ascending order). This create a slight dependence on the labelling of the problem and we therefore randomly relabelled the indices of each observation between each iteration.

### 3. Simulations

Since ARM require initial states  $z^a$  and  $z^b$  to be well-defined we begin each simulation by evaluating  $S$  chains for 50 iterations using Gibbs sampling to create an initial value of  $Z$ . To avoid any unfair advantage this initialization was used for all methods. In the simulations, both ARM and SM sampling was interlaced with Gibbs sweeps. Ie. a single iteration consist of a SM or AR move followed by a full Gibbs sweep where each observation  $i = 1, \dots, n$  is updated according to eq. (7). The number of intermediate (restricted) Gibbs sweeps for SM sampling was set to  $L = 5$  in all experiments.

To evaluate the method under diverse and realistic conditions we examine a relational and mixture-type model. The first is the Bernoulli mixture model and the artificial data of the same type considered in Jain and Neal (2004), the second is the Infinite Relational Model applied to four realistic datasets. To evaluate the correctness of the method we first ran each of the 3 methods, Gibbs sampling, SM and ARM on an Infinite Relational Model described in section 3.2 applied to a simple network problem with 4 observations (vertices) and 6 edges giving a total of 16 partitions. We compared the empirical frequency obtained from the samplers (without

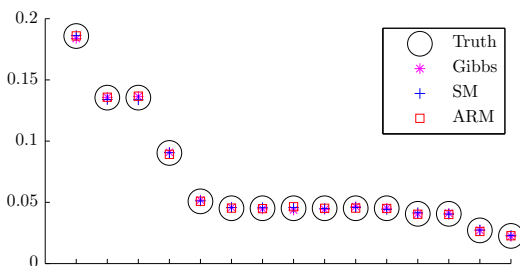


Figure 3: Gibbs, SM and ARM (the later two methods not interlaced with Gibbs sweeps) applied to the network problem where  $A_{12} = A_{13} = A_{14} = A_{34} = 1$  and otherwise 0. The plot show the true frequency obtained from evaluating the likelihood and the estimated frequency from the samplers after 160'000 iterations. All implemented methods recover the true frequency.

k	$p(A_{ij} = 1 z_j = k), i = 1, \dots, 6$					
1	.95	.95	.95	.95	.95	.95
2	.05	.05	.05	.05	.95	.95
3	.95	.05	.05	.95	.95	.95
4	.05	.05	.05	.05	.05	.05
5	.95	.95	.95	.95	.05	.05

Table 1: Mixture proportions of the Bernoulli Mixture model. For problems with more than six attributes the last column is simply copied the remaining number of times to form 3 problems with either  $d = 6, 8$  or 10 attributes.

interlacing with Gibbs sweeps for the SM and ARM samplers) over 160'000 iterations with the true value obtained by evaluating the likelihood and normalizing. The results can be seen in figure 3. Since the samplers are correlated the problem is not easily amendable to standard statistical tests, however the frequency obtained by the three methods and the true frequency obtained from the likelihood are visually in good agreement.

### 3.1 Artificial Data

The Bernoulli Mixture Model for a  $d$  feature  $\times n$  observations matrix  $A$  corresponds to the generative process and likelihood term of the form

$$\begin{aligned}
 z &\sim \text{CRP}(\alpha), & \theta_{ik} &\sim \text{Beta}(\beta_0^+, \beta_0^-), \quad i = 1, \dots, d, k = 1, 2, \dots \\
 A_{ij} &\sim \text{Bernoulli}(\theta_{iz_j}) & \log p(A|z) &= \sum_{\substack{k=1, \dots, K \\ i=1, \dots, d}} \log \frac{B(N_{ik}^+ + \beta_0^+, N_{ik}^- + \beta_0^-)}{B(\beta_0^+, \beta_0^-)}
 \end{aligned}$$

where  $N_{ik}^+ = \sum_{j \in z(k)} A_{ij}$ ,  $N_{ik}^- = \sum_{j \in z(k)} (1 - A_{ij})$  and  $\beta_0^+ = \beta_0^- = \alpha = 1$ . We generated artificial data as described by Jain and Neal (2004). The data was composed by dividing  $n = 100$  observations into  $K = 5$  components each of size 20. Each component had a variable number  $d$  of attributes such that the probability an observation assigned to a component  $k$  would have a particular attribute is given in table 1.

As the number of features  $d$  grow the observations assigned to the true blocks  $k = 1, 2, 3$  and  $k = 4, 5$  become harder to distinguish from each other and we consider three experiments (Example 1,  $d = 6$ , Example 2,  $d = 8$  and Example 3,  $d = 10$ ). Similar to Jain and Neal (2004) we computed the trace plot by considering the fraction of vertices contained in the largest block, the fraction contained in the largest and second-largest blocks and so on. A typical result can be seen in figure 4 for the three methods evaluated on the same problem. It should be noted that while ARM typically produced more jagged trace plots than SM (which in terms produced more jagged trace plots than Gibbs), there was a significant variability for different randomly generated problems and for some problems the number of components remained fixed at for instance 4 or 5 for all methods.

To get quantitative results we computed correlation time both for the first element of the trace plot and for the indicator function  $\delta_{z_i, z_j}$  for  $5 \times 3$  observations randomly selected from the 5 planted blocks. In the later case we report the average of the *maximum* of

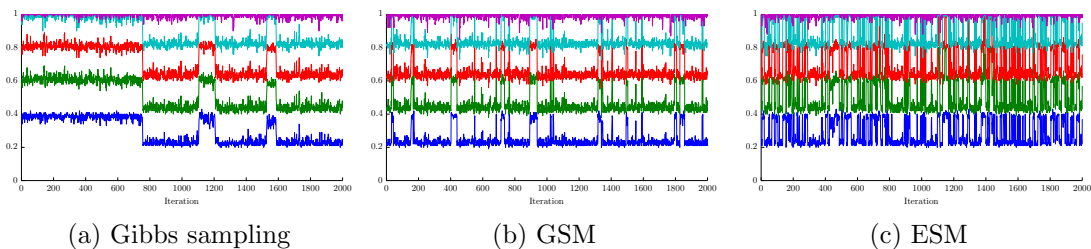


Figure 4: Example trace plots of the fraction of observations in the largest, second largest, third largest etc. block as inferred using Gibbs, SM and ARM samplers. Data was generated from Example 1 of the Bernoulli Mixture Model. For the given simulation ARM obtained better mixing than the other two and this is consistent with the other simulations, however there were significant variability in the difficulty of the (stochastically generated) sampling problems owing to their small size.

the autocorrelation time of the  $\frac{1}{2}14 \times 15$  pairs. Recall the autocorrelation time is defined as  $1 + 2 \sum_{\tau=1}^m r(\tau)$  where  $r(\tau)$  is the sample autocorrelation at the lag  $\tau$  and the sum is terminated at values beyond which the autocorrelation is close to zero (Neal, 1993).

To be consistent with Jain and Neal (2004) we used iterations rather than wall-time to compute the autocorrelation. As can be seen the total computational effort of ARM and SM are roughly equally expensive on this problem while both are about 3 times more expensive than Gibbs sampling. All results are averaged across 20 different randomly generated data sets, each being sampled by  $S = 8$  different chains evaluated for  $T = 2000$  iterations using standard settings.

The results in table 2 show quite large variations. This is mainly due to the stochastic nature of the generated data, however it makes comparison with Jain and Neal (2004) difficult since those results are only based on a single simulation evaluated for half as many iterations and with autocorrelation computed between only a single pair of observations. Our results are however consistent with their conclusion in showing split-merge result in significantly lower autocorrelation times (though these need to be seen in the light of the higher computational effort) than Gibbs sampling while ARM seem to perform better than both methods on average.

To limit the effect of the variability in the data we plotted the estimated autocorrelation times found by ARM vs. those found using SM or Gibbs in a 2D scatter plot. To reduce the number of points we have shown the mean across the  $S$  samples on the same data, see figure 6b. This scatter plot illustrate the variability in the autocorrelation times and indicate significant improvement of ARM over the other methods.

### 3.2 Relational Modelling

Our second example is the Infinite Relational Model (IRM) of Kemp et al. (2006), a non-parametric extension of a Potts-type spin model to the case of an unbounded number of

	Method	Autocorrelation		Normalized iterations
		Trace	Indicator	
Example 1	Gibbs	116.5(939)	91.6(889)	2000
	SM	27.8(122)	23.2(103)	6741(468)
	ARM	14.9(77)	13.6(71)	6712(47)
Example 2	Gibbs	165.4(1656)	131.3(1359)	2000
	SM	27.9(149)	26.4(149)	6855(482)
	ARM	8.5(50)	9.2(49)	6731(59)
Example 3	Gibbs	88.9(1374)	99.3(1327)	2000
	SM	19.3(171)	30.4(254)	7503(185)
	ARM	10.8(125)	14.7(140)	6705(44)

Table 2: Artificial data simulation results for the Bernoulli Mixture Model. The methods was evaluated on the simulated data from Example 1-3. The ARM method find significantly lower autocorrelation time both for the trace plot of the fraction of observations in the largest block and as measured by co-occurrence of observations to the same blocks. The normalization implies the autocorrelation times can be compared directly.

partitions. For symmetric network data the generative process and log likelihood becomes

$$\begin{aligned}
z &\sim \text{CRP}(\alpha) & \theta_{\ell m} &\sim \text{Beta}(\beta_0^+, \beta_0^-), \quad 1 \leq m \leq \ell \\
A_{ij} &\sim \text{Bernoulli}(\theta_{z_i z_j}) & \mathcal{L}(z) &= \sum_{1 \leq k < k' \leq K} \log \left( \frac{B(N_{kk'}^+ + \beta_0^+, N_{kk'}^- + \beta_0^-)}{B(\beta_0^+, \beta_0^-)} \right)
\end{aligned}$$

where  $N_{kk'}^+ = \sum_{i \in z(k), j \in z(k'), i \neq j} A_{ij} / 2^{\delta_{kk'}}$ ,  $N_{kk'}^- = \sum_{i \in z(k), j \in z(k'), i \neq j} (1 - A_{ij}) / 2^{\delta_{kk'}}$ . We again fixed all parameters to one. While this model is formally similar to the Bernoulli Mixture model the coupling of all components through  $\theta$  make inference more challenging.

### 3.3 Choice of data

Consider a simple data set constructed by planting  $K$  equally-sized communities in a network of  $n = dK$  vertices, such that the edge-probability between edges inside a community is higher than the edge probability between edges in different communities. Assuming recovery is possible we expect the sampler to quickly find a number  $K' < K$  communities and slowly split communities until the sampler converge at around  $K$  communities. Since each of the  $K'$  communities is (roughly) comprised of the union of one or more of the  $K$  smaller communities we can expect split-merge moves to function well especially if the community structure is clearly defined. It was data with a well-defined partition structure which was considered in the previous section.

As another extreme, consider the case of a regular  $D$ -dimensional grid with a translation-invariant boundary. In this setting it is not unrealistic to assume the average size/number of communities will be roughly constant, and accordingly Split-Merge may have very low accept rate since it assumes both the number of components and their size change. Since



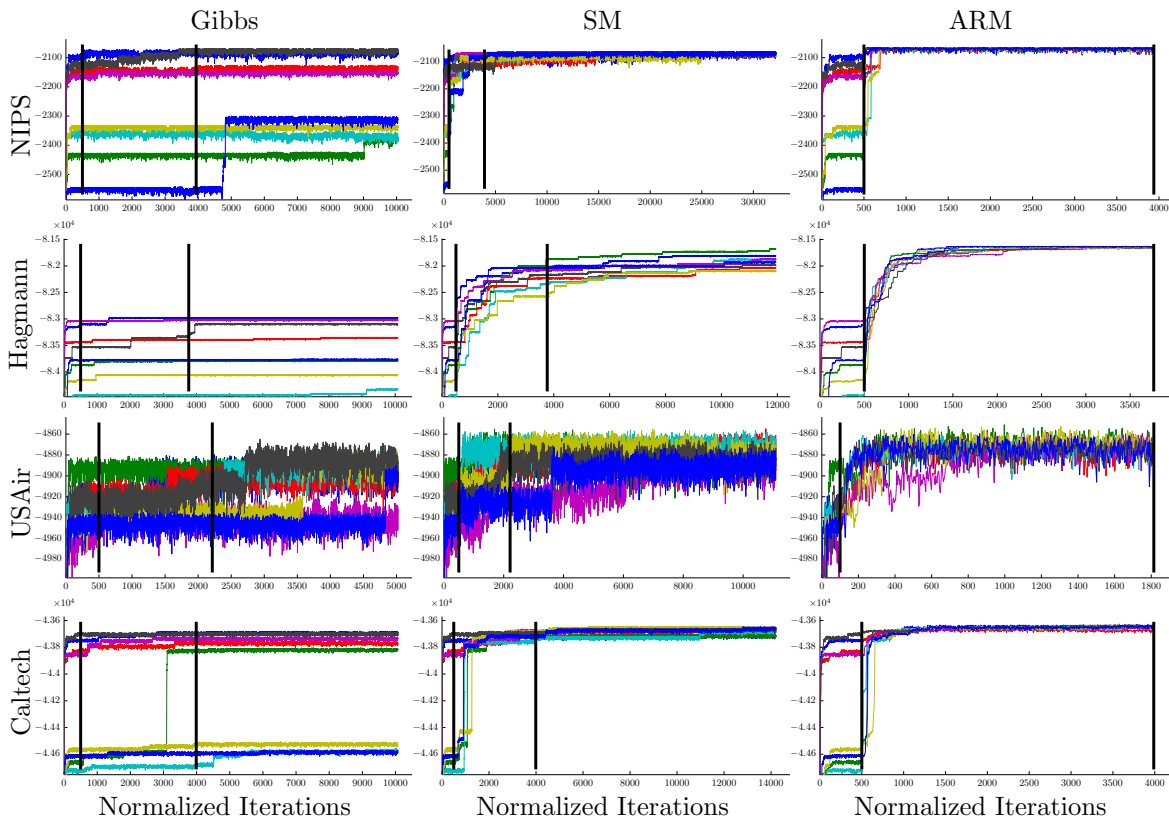


Figure 5: Trace plots of Log likelihood for Gibbs (left column), SM (middle column) and ARM sampling (right column) for all four network datasets. All simulations based on running  $S = 8$  chains using Gibbs sampling for 500 iterations, then continuing using Gibbs or SM for up to 10000 iterations or ARM for 1000 iterations. The  $x$ -scale is in normalized Gibbs iterations such that the space between the vertical black lines represent the same computational effort. The ARM method require significantly less effort to reach the same value of likelihood compared to the other methods.

the problem is translation invariant this will lead to poor mixing. While the grid provide a degenerate example, it has been shown under a wide range of conditions that Potts models on random graphs will be characterized by an exponential number of different, overlapping partitions of roughly the same likelihood and not the nested, well-defined partitions favorably to Split-Merge sampling (Borgs et al., 1999).

In addition to these structures realistic network data may contain skewed degree distribution, core-periphery structure and small-world properties. It is difficult to assess the importance of these effects, and we therefore compare the sampling methods on real network data and use downsampling to adjust the difficulty of the sampling problem. The networks considered here is the *NIPS* coauthor network ( $n = 234$ ) (available at <http://www.cs.nyu.edu/~roweis/data.html>), the USAir network ( $n = 332$ ) of US cities connected by flight routes in 1997 (available at <http://vlado.fmf.uni-lj.si/pub/networks/data>), Caltech ( $n = 769$ ), the Caltech36 social network from the Facebook100 dataset (avail-

able at <http://datahub.io/dataset/facebook100>) and the *Hagmann* structural brain network ( $n = 988$ ) of the number of fiber tracts between 998 brain regions as estimated by tractography from diffusion spectrum imaging across five subjects (see Hagmann et al. (2008)).

All networks were prepared by symmetrizing adding the transpose, tresholding at 0 and removing diagonal elements. To downsample a network from  $n$  to  $m$  vertices we sort the vertices according to the degree and label index (in descending order) and retain the  $m$  first elements.

### 3.4 Considerations for comparison

Since ARM is about three times more expensive than a Gibbs sweep the raw number of iterations does not provide a fair point of comparison. A direct comparison of wall time showed high variance in the same problem due to differences in hardware and load, however, the cost of a single (full) Gibbs sweep can be considered equal for all samplers, and we therefore report the computational effort in units of Gibbs sweeps to obtain a standardized computational cost that allows for direct comparisons. For instance the number of standardized iterations for an ARM sampler evaluated for 1000 iterations was computed to be

$$\text{Standardized Iterations} = 1000 \times \frac{\text{Total time spend}}{\text{Time spend on Gibbs iterations}}.$$

As a further point the  $S$  chains obtained by ARM are weakly dependent through the sharing of transition kernels, and comparison to a SM sampler which gives  $S$  *independent* estimates of the posterior at the same computational cost may be unfair. To account for this effect all comparisons between chains in the following sections is performed across different restarts and provide an unbiased estimate of the performance.

### 3.5 Results - Full datasets

To give a broad overview of the methods we evaluated Gibbs, SM and ARM on the full datasets and plotted the  $\log$  of the joint likelihood as a function of standardized iterations, see figure 5. All simulations were obtained by running 8 separate chains with Gibbs sampling for 10000 iterations, and using the state of the chains after 500 iterations to initialize the SM chain (which was evaluated for an additional 9500 iterations) and the ARM sampler which was evaluated for an additional 1000 iterations. The vertical black lines are meant as a visual guide to indicate the same computational efforts.

The most striking feature is the consistent poor behavior of Gibbs sampling compared to the other sampling methods. When comparing ARM and SM sampling, notice even for the smallest network - the NIPS network of only 234 vertices, most SM chains only converge after many thousands of iterations whereas ARM converged after a few tens of iterations. This result is surprising since block-type models are routinely applied to problems much larger than the NIPS network. This behavior was robust across different restarts of both methods.

For the larger networks ARM obtain higher values of the likelihood more consistently than SM, sometimes with a significant margin as in the case of the Hagmann network, however, both methods did not converge except on the NIPS network. Notice, for Gibbs

	Scale	GR- $\hat{R}$ ( $T = 2000$ )			GR- $\hat{R}$ ( $T = 1000$ )		
		Gibbs	SM	ARM	Gibbs	SM	ARM
NIPS	1	20.46(1269)	2.50(153)	1.04(2)	25.40(1090)	3.24(272)	1.01()
	0.9	23.28(765)	2.35(74)	1.00()	22.17(764)	2.74(44)	1.00()
	0.8	11.22(278)	2.75(111)	1.01()	13.26(452)	2.89(160)	1.01(1)
Hagmann	0.4	15.46(1363)	4.24(181)	1.76(5)	15.96(1397)	4.39(237)	1.77(4)
	0.3	9.96(678)	1.69(39)	1.06(7)	10.62(680)	1.81(46)	1.13(16)
	0.2	3.78(239)	1.02(1)	1.01()	4.54(229)	1.19(23)	1.02(1)
USAir	0.9	2.49(124)	1.30(29)	1.02(2)	2.72(100)	1.82(113)	1.01()
	0.8	2.41(107)	2.26(99)	1.00()	2.45(113)	2.19(90)	1.00()
	0.7	1.04(3)	1.08(4)	1.02(2)	1.04(3)	1.12(8)	1.03(2)
Caltech	0.8	32.13(884)	2.98(101)	1.18(31)	23.65(1666)	2.37(154)	1.26(32)
	0.7	4.32(249)	2.97(144)	1.01(1)	9.42(940)	2.71(141)	1.04(3)
	0.6	8.49(462)	2.44(76)	1.01(1)	7.28(335)	2.56(82)	1.03(2)

Table 3: Results for varying the downsampling (scale is the fraction of remaining observations) as well as the number of (normalized) iterations. Gelman-Rubin (GR) statistics are computed on the trace plot of the likelihood for 4 different restarts and  $S = 8$ . Half the samples are discarded as burning. Gelman-Rubin statistics less than 1.1-1.2 are normally considered compatible with mixing and ARM obtain lower GR values in fewer (normalized) iterations compared to the other methods.

sampling each chain will behave in a stationary fashion for long stretches of time until it jump to a more favorable plateau. Since the chain appears to be converged during this time the most principal comparison seems to be between-chains statistics. We have focused on the Gelman-Rubin potential scale reduction factor  $\hat{R}$  which attempt to quantify the likelihood samples from different chains came from the same distribution. Ideally  $\hat{R}$  should be near 1, and values lover than 1.1 or 1.2 is generally considered consistent with convergence (Brooks and Gelman, 1998). We experimented on different quantities used to compute the GR statistics but settled on the joint log likelihood which seem to give a reasonable discriminative power. As described in section 3.4 we always compute GR statistics between different restarts creating an unbiased estimate and while we report simulation time in terms of number of ARM iterations we compute the statistics for the other method based on a similar number of normalized Gibbs iterations as described in section 3.4.

Simulation results on downsampled networks is available in table 3 for simulations where  $S = 8$  and the GR-statistics are computed based on 4 restarts. Half the samples were discarded as burnin in the experiments.

Many of the networks required quite aggressive downsampling (the level was selected based on a coarse search as the highest value where at least one method have a favorable GR statistic) and the GR statistics show considerable variance. Somewhat counter-intuitively, a chain which is far from convergence will tend to have an increasing value of the likelihood during the sampling. Since this will inflate the within-chain variance, it will tend to lower the GR-statistics, and this effect seems partially responsible for the large variance. As can be seen none of the methods mix for the considered number of iterations except possibly the

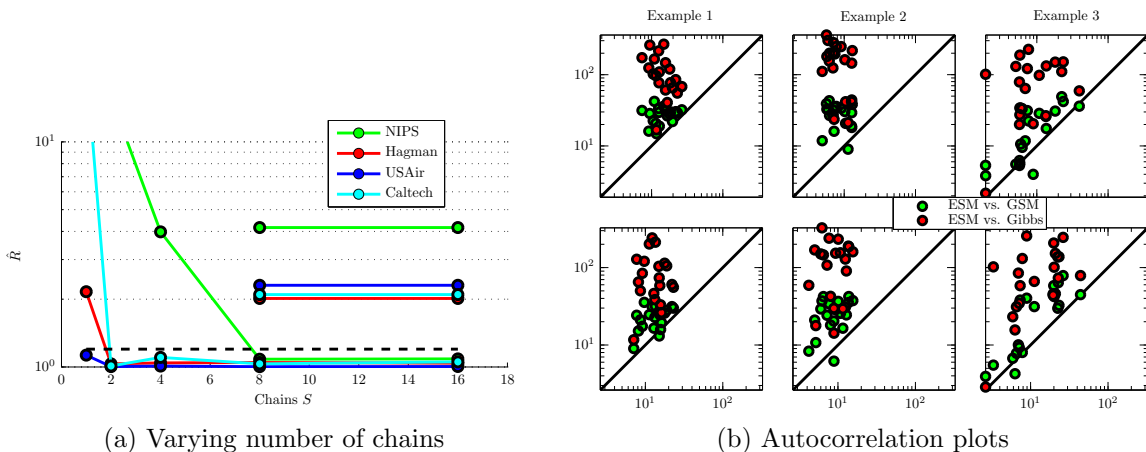


Figure 6: *Left:* Effect of varying the number of chains on  $\hat{R}$ . For comparison we show results obtained using SM sampling as a vertical line. As can be seen, ARM benefit from the use of additional chains. *Right:* Scatter plots of autocorrelation time for the trace of the first block (top row) and for the indicators (bottom row) for the Bernoulli mixture model as described in the text. Points above the diagonal line indicate ARM perform better than the other method. As can be seen all methods have significant variances, however, there is a clear trend towards ARM performing better than SM for the considered problems.

NIPS network. After downsampling Gibbs sampling continue to perform very poorly and ARM consistently perform better than SM sampling. Interestingly doubling the number of iterations from 1000 to 2000 does not seem to have a large effect.

### 3.6 Effects of variation of the method

The proposed method consist of several components such as the use of multiple parallel chains and the manner in which blocks of observations are reassigned to create bolder moves. To assess the impact of these ideas we will briefly discuss some variations of the method.

#### 3.6.1 VARYING IN THE NUMBER OF CHAINS

An important question is how well the method benefit from improved parallelism. The simplest experiment is to vary the number of parallel chains  $S$  between 2 and 16 for relevant levels of downsampling chosen from table 3. The result can be seen in figure 6a. As can be seen the method typically benefit from higher values of  $S$  and for very small values of  $S$  ARM sampling sometimes perform worse than Split-Merge sampling. This underperformance is not necessarily due to the method producing worse proposals since the additional computational cost of ARM means that it will consider significantly less proposal splits.

#### 3.6.2 USE OF BLOCKS AND INITIALIZATION

To investigate the use of blocks when performing splits as well as how the method benefit from selecting the "correct" initial variables  $i, j$  and initial splits, we proposed two variants

	Scale	Gibbs	SM	bSM	sARM	ARM
NIPS	1	24.88(823)	2.43(97)	1.12(15)	2.41(277)	1.09(3)
Hagmann	0.25	2.86(88)	1.99(84)	1.65(40)	1.28(33)	1.08(9)
USAir	0.8	2.32(144)	1.94(105)	1.54(74)	1.60(102)	1.01(1)
Caltech	0.7	20.58(1396)	3.41(119)	1.71(119)	1.61(96)	1.11(9)

Table 4: Results for variations over SM and ARM samplers on downsampled network data. The numbers reported is the GR statistic of the log joint likelihood based on 1000 iterations (half discarded as burnin) for  $S = 8$  and 4 restarts. GR scores lower than 1.1-1.2 are usually considered consistent with mixing. The results indicate that while various features of the ARM sampler give some benefit individually the full method perform better than the variations considered individually.

of the SM and ARM methods. For the SM method, we choose the same initialization as in ARM, i.e., instead of randomly distributing the elements contained in  $z_i \cup z_j \setminus \{i, j\}$  between the two new communities in the launch state we now select the same quintet  $(i, j, z^a, z^b)$  as the ARM method and initialize the split configuration in the launch state in a similar way as ARM. Specifically, we set  $z_i^{(l)} = c_i$  and  $z_j^{(l)} = c_j$  where  $c$  is the coarsest common refinement between  $z, z^a$  and  $z^b$ . We dub this method bSM.

For the ARM sampler we considered a variant very similar to sRM described in algorithm 2. However, in order to create a fair comparison we still select  $i, j$  and the initial partitions  $z^a, z^b$  the same way as for regular ARM and again attempt the informed initialization where  $z_i^* = c_i$  and  $z_j^* = c_j$ . This variant is dubbed sARM. Both new methods are interlaced with Gibbs sweeps and initialized similar to SM and ARM.

We compared these methods on downsampled networks for the equivalent of 1000 ARM iterations (half discarded as burnin) based on 4 restarts and  $S = 8$ . The results can be seen in table 4.

Again ARM seem to perform either on par or better than SM, bSM and Gibbs while being slightly better than sARM, the magnitude seem quite dependent on the dataset. To better understand the behavior of the different methods we also computed the mean of the accept rate ( $\times 100$ ) from eq. (8) for all methods except Gibbs sampling. The clearest pattern is SM has accept rate less than half a percent for all datasets and the use of past states to guide the initialization only seem to double this number. While the accept rate of ARM is larger than for sARM on all datasets, it is much higher for the NIPS and Hagmann networks while being fairly low for USAir.

## 4. Discussion

Due to the explicit block-structure present in the Bernoulli mixture model examples we expected these to provide an ideal sampling problem for the split-merge method. While all samplers could easily solve the problem (computing GR statistics of the log joint likelihood gave results very near 1 in all instances), it seems that ARM and SM perform better than Gibbs sampling and ARM performs better still than SM.

	Scale	SM	bSM	sARM	ARM
NIPS	1	0.48(3)	1.01(8)	1.37(14)	19.41(128)
Hagmann	0.25	0.14(1)	0.37(6)	1.08(27)	13.56(242)
USAir	0.8	0.04(2)	0.11(4)	0.35(5)	4.10(19)
Caltech	0.7	0.09(4)	0.32(8)	0.84(9)	6.82(43)

Table 5: Acceptrate $\times 100$  for variations of SM and ARM samplers on downsampled network data. As can be seen, the variation in accept rate can be very dramatic when comparing SM to ARM.

These results are convolved by the high variability in autocorrelation times. Consider for instance the autocorrelation of the trace plot as seen in figure 4. While Gibbs sampling no doubt has a high autocorrelation time, a problem where Gibbs sampling perform slightly worse and stayed in a single mode for the entire duration of the simulation would counterintuitive have a very low autocorrelation. This problem is aggravated by the stochastic nature of the test data and gave very high variance in the reported quantities. Still, since the autocorrelation is a bounded quantity, higher variance can be expected to be correlated with worse performance and there is a distinct trend towards ARM having lower mean autocorrelation and lower variance than SM. This conclusion is reinforced by figure 6b which shows very high between-dataset variability but a clear trend towards most points being above the diagonal line indicating better performance by ARM.

For the more realistic network-data examples table 5 of averaged accept rates provide an interesting comparison between the methods. The most visible feature is the abysmal accept rate of split-merge sampling, typically a few tens of a percent to at most half a percent. It is interesting that despite such low accept rates split-merge sampling still provides improvements over Gibbs sampling for all datasets (see table 4 and figure 5). The low accept rate is likely due to the nature of split-merge sampling requiring the number of components to grow or shrink by one, this view is supported in that the variant of SM sampling, bSM, which attempt to reuse past information of chains to provide the same initialization as ARM only has about twice the accept rate compared to the full ARM method which has accept rates up to 10-20% for the NIPS and Hagmann network.

Focusing on the role of moving blocks, we see in particular for the networks with high accept rate (NIPS and Hagmann) movement of blocks cause the accept rate to increase by a factor 5-10 (compared to sARM). While the gains are more modest for the other problems they remain a consistent feature. As a corollary sARM outperform bSM in terms of accept rate uniformly. Taken together with the increased performance of ARM compared to SM, Gibbs, bSM and sARM in table 4 this support the idea that moves that allow exchange of observations with blocks other than those presently being split or merged should be an important guiding feature in creating future samplers for partition-based problems, and movement of blocks boost the accept rate.

ARM seem to perform better than SM in nearly all settings, see table 3, and for most datasets the increase in performance does not seem to be easily matched by simply increasing the simulation time of SM. The difference in behavior is perhaps most striking when simply visualizing the trace plots (see figure 5) since when translating the performance

into numerical quantities the methods which has not converged show high variance. However, the trace plots indicate that SM can be considered tens of times slower than ARM on realistic datasets.

The use of past states not only has a positive effect on the accept rate but also help exploration. This is indicated in figure 6a which suggests the minimum number of parallel chains to be used should be about 4-8, but that the method seem to scale favorably with increased parallelism. This is a significant result in light of the current trend where computers scale in parallelism rather than speed.

## 5. Conclusion and Further Research

We have presented a novel method for sampling partition-based models. The proposed method evaluates an ensemble of chains in parallel and use their current and past states to construct an adaptive transition kernel. While the transitions will either split or merge two selected observations, in contrast to standard Split-Merge operations the number of blocks may increase, decrease or stay the same regardless of the type of operations.

Our simulations indicate the method has superior performance over Split-Merge or Gibbs sampling for the Bernoulli Mixture model and the Infinite Relational Model on both artificial and real data, and by considering variants of the method we have shown the particular use of past states as well as the extended space of available transitions both contribute towards the methods performance, and the use of multiple chains play a crucial role in providing exploration and increased accept rate. The increased parallelism in modern computing methods can take advantage of this capacity.

It is worth emphasizing the two key components in the present method, the use of past states to inform the current moves and the use of bolder proposals can likely be furthered with some modifications to the above framework. Some proposals for future research could be softening the requirement that only two observations,  $i, j$ , are forcibly split or merged, examining the role of using more than two past states in constructing the transition kernel or allowing movement of blocks of variables from outside the initial states  $i, j$ .

The extension to non-conjugate models is fairly straight forward using techniques such as Algorithm 8 of Neal (2000). However, a procedures more in line with the current work could be to use the value of the (non-conjugate) parameters obtained from the initial states  $z^a$  and  $z^b$  rather than draws from the prior to inform the proposal distributions.

Despite the wide application of partition-based Bayesian models the construction of good transition kernels remain a largely under-explored area. Our results suggest transitions involving more than two blocks are of vital importance to obtain high accept rates, adding to the problem of how to propose near-equilibrium states. Our current proposal attempts to alleviate the later issue using information from past states rather than restricted Split-Merge operations. While there is no doubt many variations of how this information could be gathered or put the *need* to include such information in the proposal distribution seem to be a robust feature.

## Acknowledgments

This project was funded in part by the Lundbeck Foundation.

## References

- Charles E Antoniak. Mixtures of dirichlet processes with applications to bayesian nonparametric problems. *The annals of statistics*, pages 1152–1174, 1974.
- Yves F Atchadé and Jeffrey S Rosenthal. On adaptive markov chain monte carlo algorithms. *Bernoulli*, 11(5):815–828, 2005.
- Adrian Barbu and Song-Chun Zhu. Generalizing swendsen-wang to sampling arbitrary posterior probabilities. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(8):1239–1253, 2005.
- Christian Borgs, Jennifer T Chayes, Alan Frieze, Jeong Han Kim, Prasad Tetali, Eric Vigoda, et al. Torpid mixing of some monte carlo markov chain algorithms in statistical physics. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 218–229. IEEE, 1999.
- Stephen P Brooks and Andrew Gelman. General methods for monitoring convergence of iterative simulations. *Journal of computational and graphical statistics*, 7(4):434–455, 1998.
- Gilles Celeux, Merrilee Hurn, and Christian P Robert. Computational and inferential difficulties with mixture posterior distributions. *Journal of the American Statistical Association*, 95(451):957–970, 2000.
- David B Dahl. An improved merge-split sampler for conjugate dirichlet process mixture models. *Technical R eport*, 1:086, 2003.
- Michael D Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the american statistical association*, 90(430):577–588, 1995.
- Thomas S Ferguson. A bayesian analysis of some nonparametric problems. *The annals of statistics*, pages 209–230, 1973.
- Peter J Green and Sylvia Richardson. Modelling heterogeneity with and without the dirichlet process. *Scandinavian journal of statistics*, 28(2):355–375, 2001.
- P Hagmann, L Cammoun, X Gigandet, R Meuli, C J Honey, V J Wedeen, and O Sporns. Mapping the structural core of human cerebral cortex. *PLoS biology*, 6(7):e159, 2008.
- W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.
- Sonia Jain and Radford M Neal. A split-merge markov chain monte carlo procedure for the dirichlet process mixture model. *Journal of Computational and Graphical Statistics*, 13(1):158–182, 2004. doi: 10.1198/1061860043001. URL <http://amstat.tandfonline.com/doi/abs/10.1198/1061860043001>.



- Charles Kemp, Joshua B Tenenbaum, Thomas L Griffiths, Takeshi Yamada, and Naonori Ueda. Learning Systems of Concepts with an Infinite Relational Model. In *AAAI*, pages 381–388, 2006.
- John W Lau and Peter J Green. Bayesian model-based clustering procedures. *Journal of Computational and Graphical Statistics*, 16(3):526–558, 2007.
- Steven N MacEachern. Estimating normal means with a conjugate style dirichlet process prior. *Communications in Statistics-Simulation and Computation*, 23(3):727–741, 1994.
- Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21:1087, 1953.
- Radford M Neal. Probabilistic inference using markov chain monte carlo methods. 1993.
- Radford M Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of computational and graphical statistics*, 9(2):249–265, 2000.
- RadfordM. Neal. Bayesian mixture modeling. In C.Ray Smith, GaryJ. Erickson, and PaulO. Neudorfer, editors, *Maximum Entropy and Bayesian Methods*, volume 50 of *Fundamental Theories of Physics*, pages 197–211. Springer Netherlands, 1992. ISBN 978-90-481-4220-0. doi: 10.1007/978-94-017-2219-3\_14. URL [http://dx.doi.org/10.1007/978-94-017-2219-3\\_14](http://dx.doi.org/10.1007/978-94-017-2219-3_14).
- Jim Pitman et al. Combinatorial stochastic processes. Technical report, Technical Report 621, Dept. Statistics, UC Berkeley, 2002. Lecture notes for St. Flour course, 2002.
- Gareth O. Roberts and Jeffrey S. Rosenthal. Coupling and ergodicity of adaptive markov chain monte carlo algorithms. *J. Appl. Probab.*, 44(2):458–475, 03 2007. doi: 10.1239/jap/1183667414. URL <http://dx.doi.org/10.1239/jap/1183667414>.
- Gareth O Roberts and Jeffrey S Rosenthal. Examples of adaptive mcmc. *Journal of Computational and Graphical Statistics*, 18(2):349–367, 2009.
- Jayaram Sethuraman. A constructive definition of dirichlet priors. Technical report, DTIC Document, 1991.
- Robert H Swendsen and Jian-Sheng Wang. Nonuniversal critical dynamics in monte carlo simulations. *Physical review letters*, 58(2):86–88, 1987.
- Yee W Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Sharing clusters among related groups: Hierarchical dirichlet processes. In *Advances in Neural Information Processing Systems*, pages 1385–1392, 2004.
- Luke Tierney. Markov chains for exploring posterior distributions. *the Annals of Statistics*, pages 1701–1728, 1994.
- H. Toutenburg. Everitt, b. s.: Introduction to latent variable models. chapman and hall, london 1984. 107 pp., 9.50. *Biometrical Journal*, 27(6):706–706, 1985. ISSN 1521-4036. doi: 10.1002/bimj.4710270617. URL <http://dx.doi.org/10.1002/bimj.4710270617>.

- Z Xu, V Tresp, K Yu, and H P Kriegel. Infinite hidden relational models. In *Proceedings of the 22nd International Conference on Uncertainty in Artificial Intelligence (UAI 2006)*, 2006.
- Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. Multi-task learning for classification with dirichlet process priors. *The Journal of Machine Learning Research*, 8:35–63, 2007.